Printing count of primes in range [L – R]

There can be multiple ranges (queries)

We can use isPrime helper method with $\sqrt{n}$ complexity for each range.

$$T.C\,(\underset{\text{queries}}{\underset{\uparrow}{Q}} \times \underset{\text{Range}}{\underset{\uparrow}{(R-L+1)}} \times \underset{\text{isPrime}}{\underset{\uparrow}{\sqrt{n}}})$$

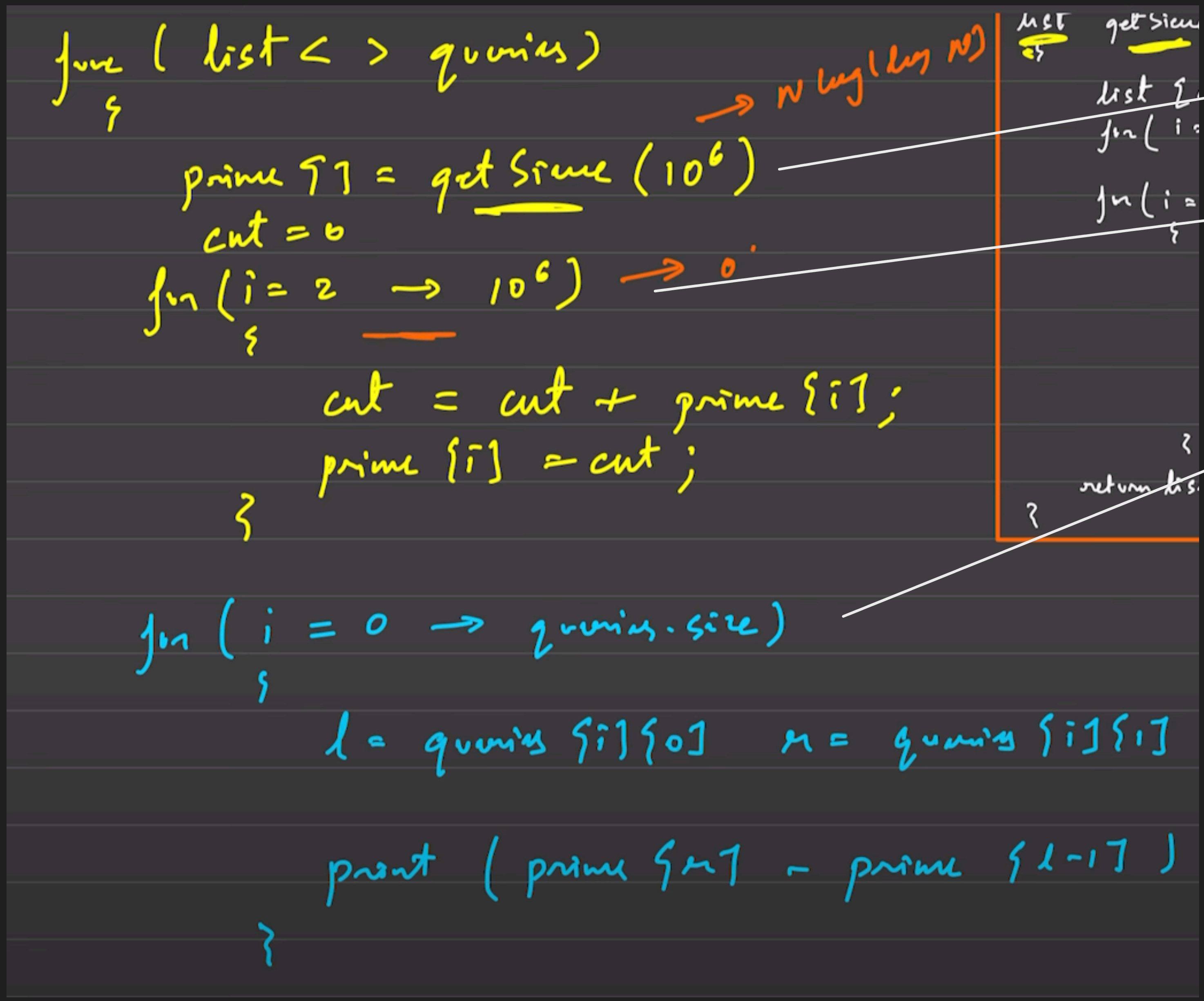
This is naive approach and not acceptable.

One step in improving T.C. is converting check of isPrime to constant by pre-computing all primes till max possible value of queries boundary. This computation takes $\log(\log n)$

Still we only improve $\sqrt{n}$ to one pre-calculation of $\log(\log n)$.

For most optimal solution we can pre-compute the number of primes till i^{th} idx. Prefix sum.
Now, for each query we can check

queryResult[i] = primeCount [R] - primeCount [L - 1] <- check boundary here



```
func ( list< > queries)
{
    prime [] = getSieve (10^6)        -> N log(log N)
    cnt = 0
  for (i = 2  -> 10^6)                -> 0
  {
        cnt = cnt + prime [i];
        prime [i] = cnt ;
  }

  for ( i = 0 -> queries.size)
  {
        l = queries [i][0]     r = queries [i][1]

        print ( prime [r] - prime [l-1] )
  }
```

$$\left.\begin{matrix} \log(\log 10^6) \\ + \\ 10^6 \\ + \\ Q \end{matrix}\right\} T.C$$